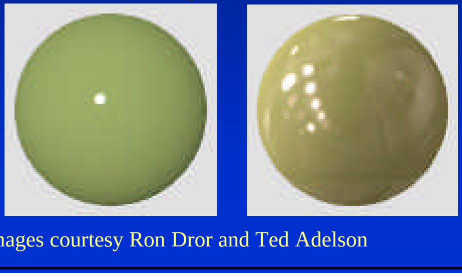
An Efficient Representation for Irradiance Environment Maps

Ravi Ramamoorthi   Pat Hanrahan

Stanford University

Natural Illumination

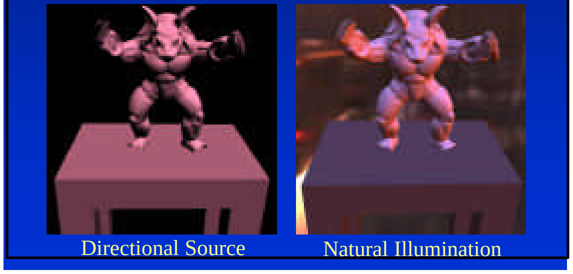
People perceive materials more easily under natural illumination than simplified illumination.



Images courtesy Ron Dror and Ted Adelson

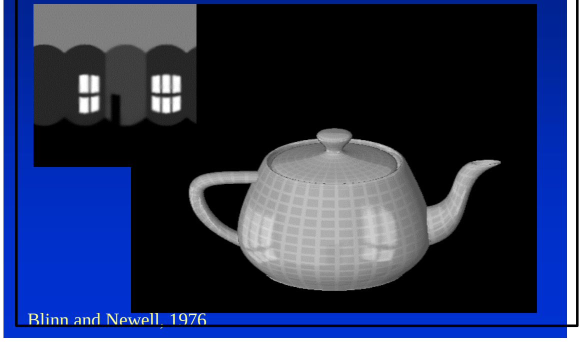
Natural Illumination

Classically, rendering with natural illumination is very expensive compared to using simplified illumination



Directional Source   Natural Illumination

Reflection Maps

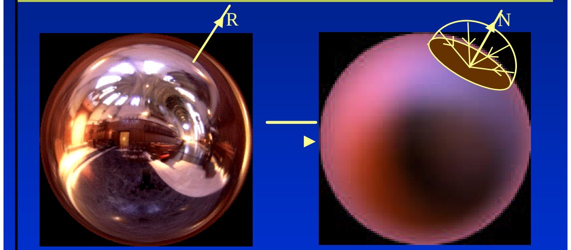


Blinn and Newell, 1976

Environment Maps

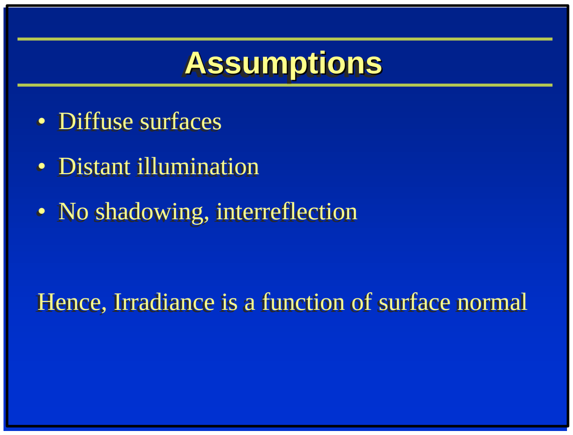
Miller and Hoffman, 1984

Irradiance Environment Maps



Incident Radiance (Illumination Environment Map)

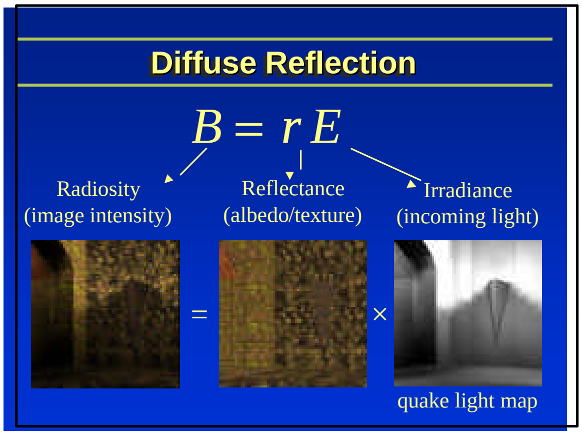
Irradiance Environment Map



Assumptions

- Diffuse surfaces
- Distant illumination
- No shadowing, interreflection

Hence, Irradiance is a function of surface normal



Diffuse Reflection

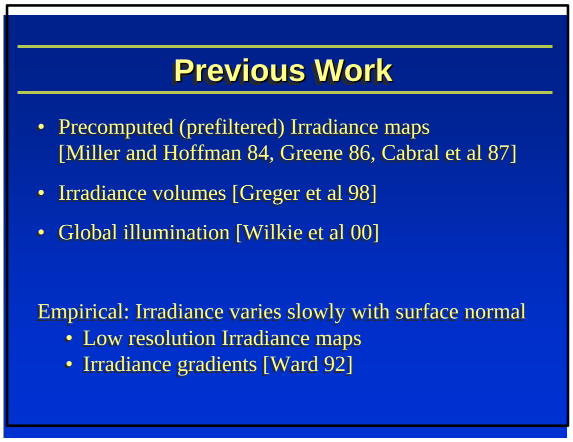
$$B = r\,E$$

Radiosity (image intensity)

Reflectance (albedo/texture)

Irradiance (incoming light)

quake light map

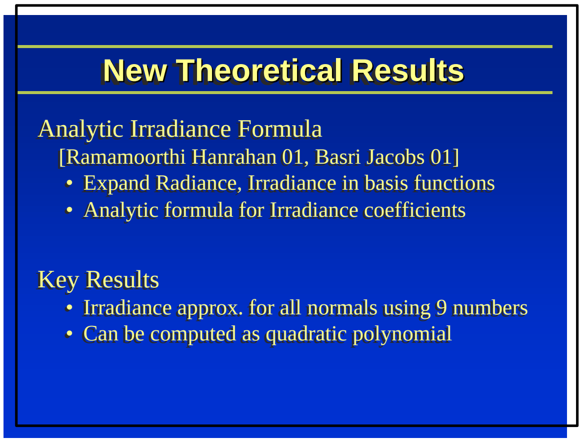


Previous Work

- Precomputed (prefiltered) Irradiance maps [Miller and Hoffman 84, Greene 86, Cabral et al 87]
- Irradiance volumes [Greger et al 98]
- Global illumination [Wilkie et al 00]

Empirical: Irradiance varies slowly with surface normal

- Low resolution Irradiance maps
- Irradiance gradients [Ward 92]



New Theoretical Results

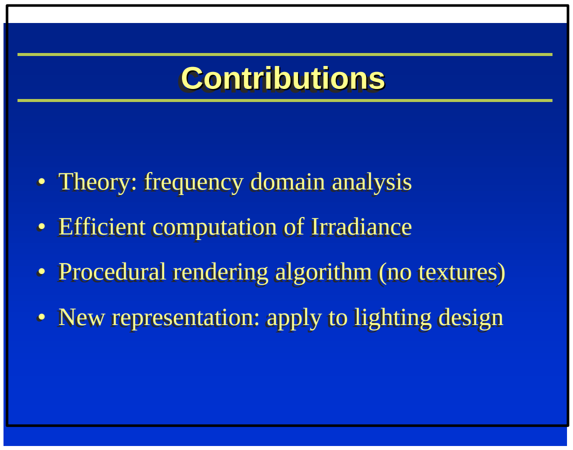
Analytic Irradiance Formula

[Ramamoorthi Hanrahan 01, Basri Jacobs 01]

- Expand Radiance, Irradiance in basis functions
- Analytic formula for Irradiance coefficients

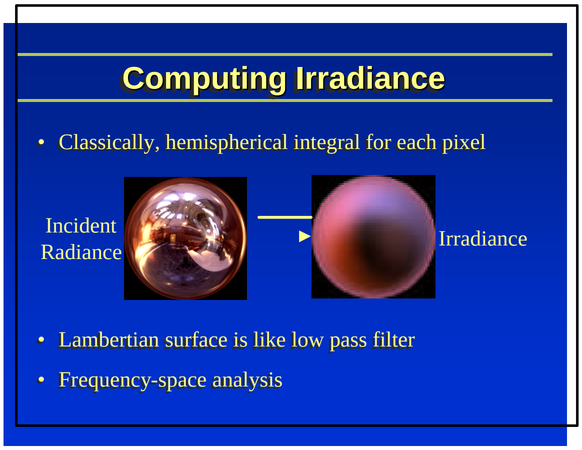
Key Results

- Irradiance approx. for all normals using 9 numbers
- Can be computed as quadratic polynomial



Contributions

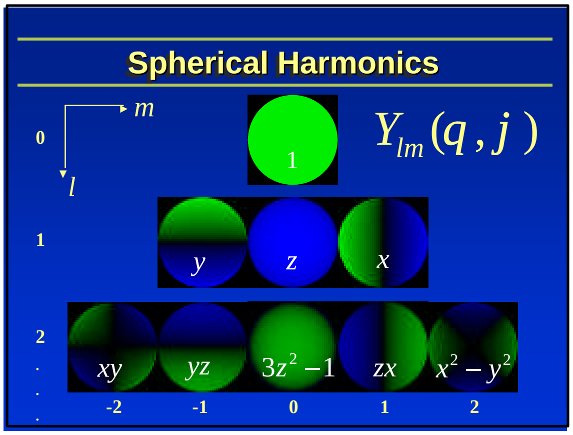
- Theory: frequency domain analysis
- Efficient computation of Irradiance
- Procedural rendering algorithm (no textures)
- New representation: apply to lighting design



Computing Irradiance

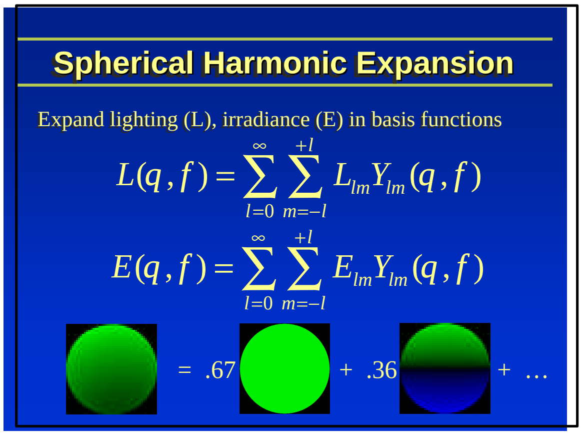
- Classically, hemispherical integral for each pixel

- Lambertian surface is like low pass filter
- Frequency-space analysis



Spherical Harmonics

$Y_{lm}(q,j)$

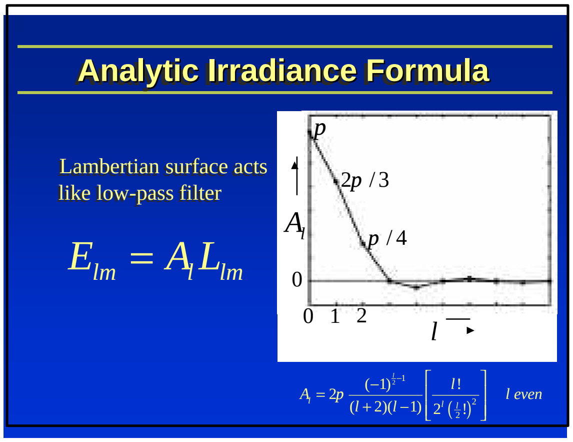


Spherical Harmonic Expansion

Expand lighting (L), irradiance (E) in basis functions

$$L(q,f)=\sum_{l=0}^{\infty}\sum_{m=-l}^{+l}L_{lm}Y_{lm}(q,f)$$

$$E(q,f)=\sum_{l=0}^{\infty}\sum_{m=-l}^{+l}E_{lm}Y_{lm}(q,f)$$

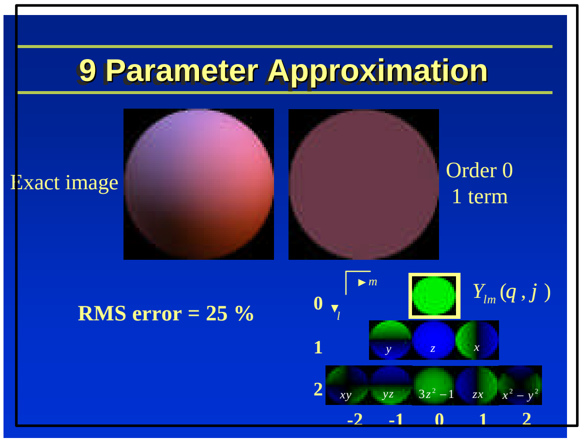


Analytic Irradiance Formula

Lambertian surface acts like low-pass filter

$$E_{lm}=A_l L_{lm}$$

$$A_l = 2p\,\frac{(-1)^{\frac{l}{2}-1}}{(l+2)(l-1)}\left[\frac{l!}{2^l\left(\frac{l}{2}!\right)^2}\right] \qquad l\ \textit{even}$$

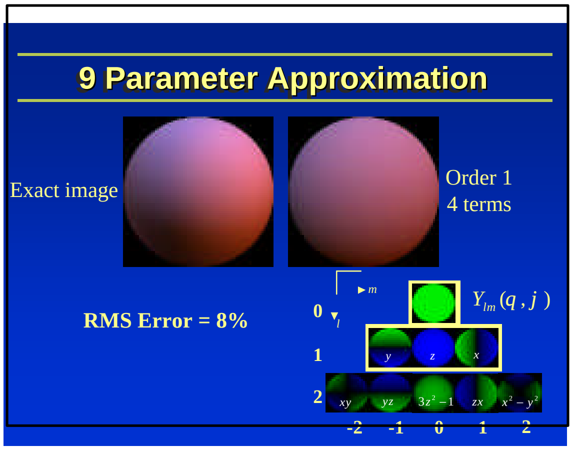


9 Parameter Approximation

Exact image

Order 0
1 term

RMS error = 25 %

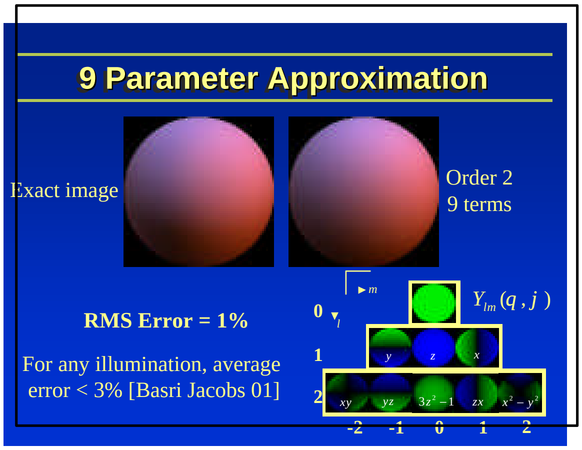


9 Parameter Approximation

Exact image

Order 1
4 terms

RMS Error = 8%



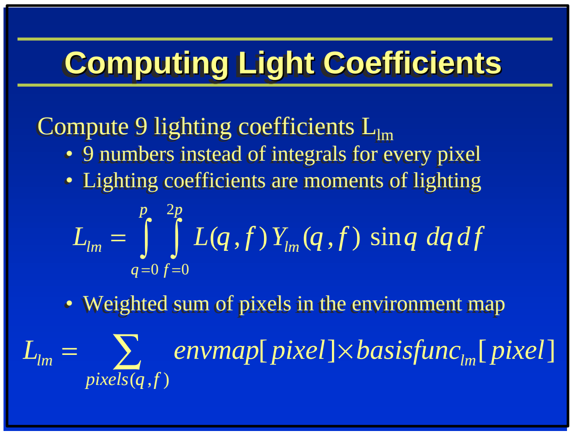
9 Parameter Approximation

Exact image

Order 2
9 terms

RMS Error = 1%

For any illumination, average error < 3% [Basri Jacobs 01]



Computing Light Coefficients

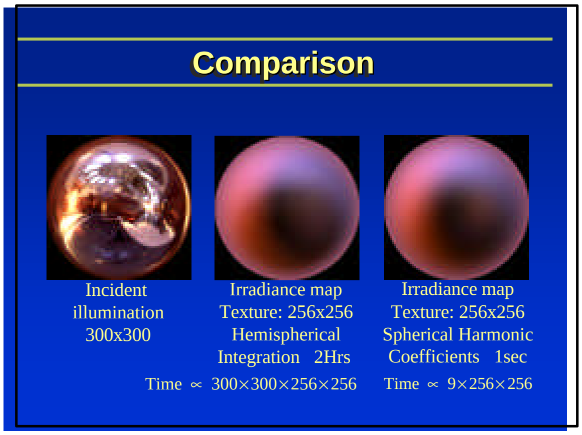
Compute 9 lighting coefficients L_{lm}

- 9 numbers instead of integrals for every pixel
- Lighting coefficients are moments of lighting

$$L_{lm} = \int_{q=0}^{p} \int_{f=0}^{2p} L(q,f)\, Y_{lm}(q,f)\, \sin q \; dq\, df$$

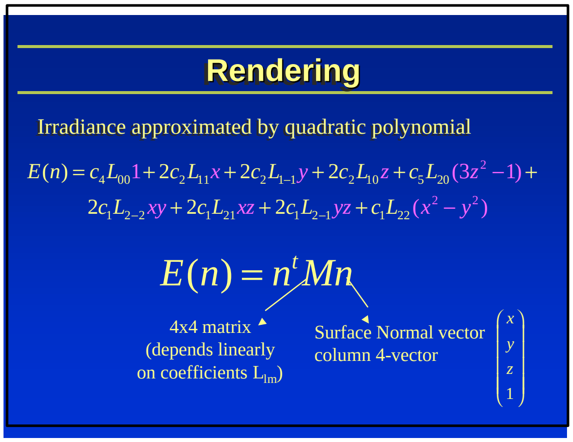
- Weighted sum of pixels in the environment map

$$L_{lm} = \sum_{pixels(q,f)} envmap[\,pixel\,] \times basisfunc_{lm}[\,pixel\,]$$



Comparison

Incident illumination 300x300	Irradiance map Texture: 256x256 Hemispherical Integration 2Hrs	Irradiance map Texture: 256x256 Spherical Harmonic Coefficients 1sec
	Time ∝ 300×300×256×256	Time ∝ 9×256×256



Rendering

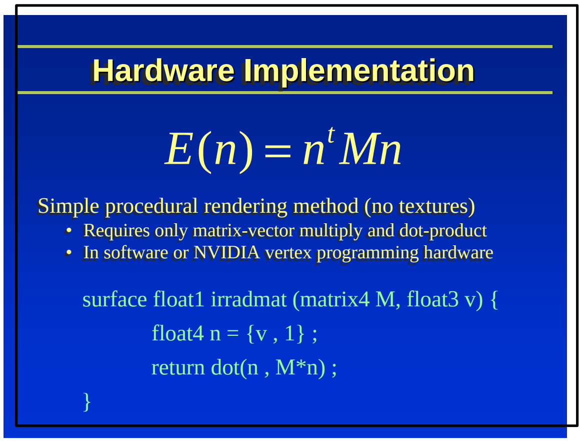
Irradiance approximated by quadratic polynomial

$$E(n) = c_4 L_{00} 1 + 2c_2 L_{11} x + 2c_2 L_{1-1} y + 2c_2 L_{10} z + c_5 L_{20}(3z^2 - 1) +$$
$$2c_1 L_{2-2} xy + 2c_1 L_{21} xz + 2c_1 L_{2-1} yz + c_1 L_{22}(x^2 - y^2)$$

$$E(n) = n^t M n$$

4x4 matrix (depends linearly on coefficients L_{lm})

Surface Normal vector column 4-vector $\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$



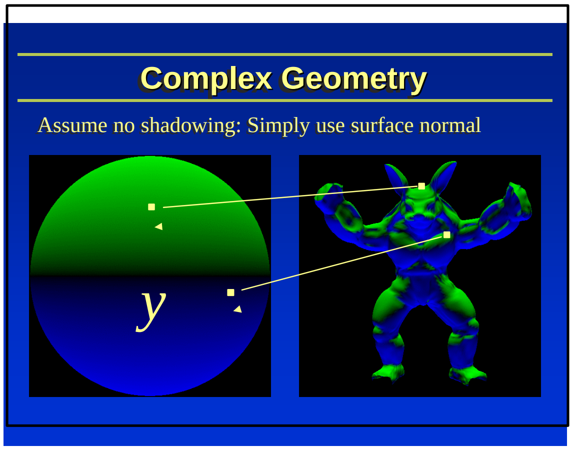
Hardware Implementation

$$E(n) = n^t M n$$

Simple procedural rendering method (no textures)

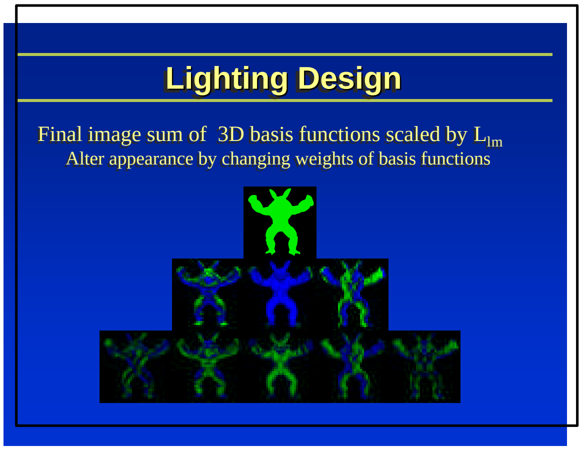
- Requires only matrix-vector multiply and dot-product
- In software or NVIDIA vertex programming hardware

```
surface float1 irradmat (matrix4 M, float3 v) {
        float4 n = {v , 1} ;
        return dot(n , M*n) ;
}
```



Complex Geometry

Assume no shadowing: Simply use surface normal

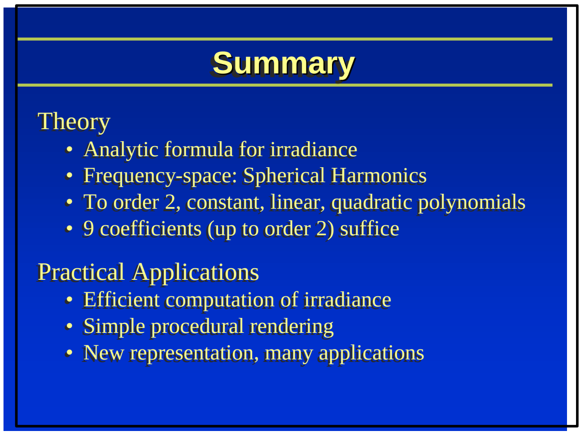


Lighting Design

Final image sum of 3D basis functions scaled by L_{lm}

Alter appearance by changing weights of basis functions

Demo



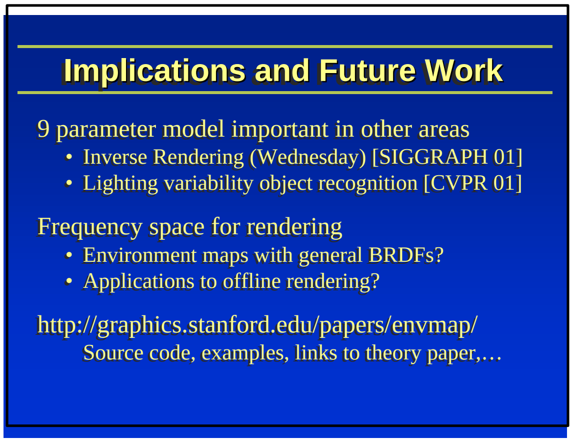
Summary

Theory

- Analytic formula for irradiance
- Frequency-space: Spherical Harmonics
- To order 2, constant, linear, quadratic polynomials
- 9 coefficients (up to order 2) suffice

Practical Applications

- Efficient computation of irradiance
- Simple procedural rendering
- New representation, many applications



Implications and Future Work

9 parameter model important in other areas

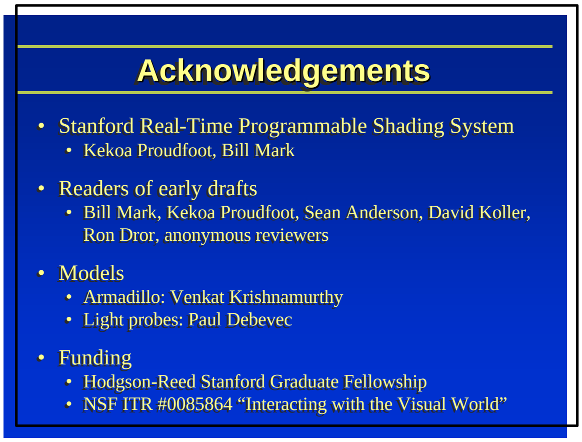
- Inverse Rendering (Wednesday) [SIGGRAPH 01]
- Lighting variability object recognition [CVPR 01]

Frequency space for rendering

- Environment maps with general BRDFs?
- Applications to offline rendering?

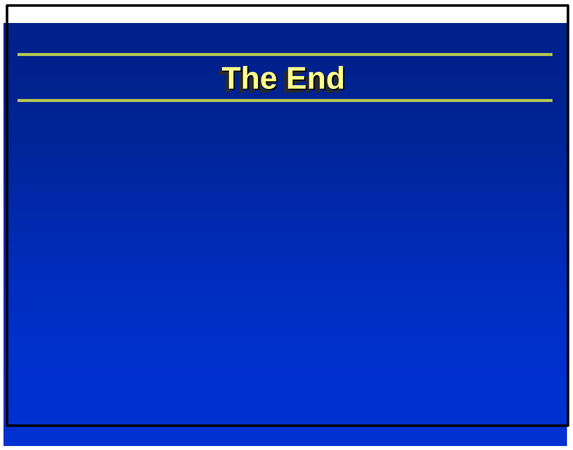
http://graphics.stanford.edu/papers/envmap/

Source code, examples, links to theory paper,…

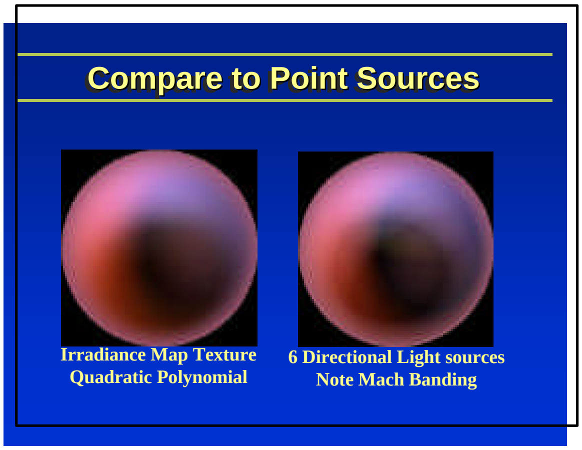


Acknowledgements

- Stanford Real-Time Programmable Shading System
  - Kekoa Proudfoot, Bill Mark
- Readers of early drafts
  - Bill Mark, Kekoa Proudfoot, Sean Anderson, David Koller, Ron Dror, anonymous reviewers
- Models
  - Armadillo: Venkat Krishnamurthy
  - Light probes: Paul Debevec
- Funding
  - Hodgson-Reed Stanford Graduate Fellowship
  - NSF ITR #0085864 "Interacting with the Visual World"



The End



Compare to Point Sources

Irradiance Map Texture
Quadratic Polynomial

6 Directional Light sources
Note Mach Banding